View metadata, citation and similar papers at core.ac.uk

brought to you by CORE

provided by Harvard University - DASH

Sentence disambiguation by a shift-reduce parsing technique

The Harvard community has made this article openly available. Please share how this access benefits you. Your story matters.

Citation	Stuart M. Shieber. Sentence disambiguation by a shift-reduce parsing technique. In Proceedings of the 21st Annual Meeting of the Association for Computational Linguistics, pages 113-118, Massachusetts Institute of Technology, Cambridge, Massachusetts, June15-17 1983.
Published Version	doi:10.3115/981311.981334
Accessed	February 17, 2015 1:24:46 PM EST
Citable Link	http://nrs.harvard.edu/urn-3:HUL.InstRepos:2265292
Terms of Use	This article was downloaded from Harvard University's DASH repository, and is made available under the terms and conditions applicable to Other Posted Material, as set forth at http://nrs.harvard.edu/urn-3:HUL.InstRepos:dash.current.terms-of-use#LAA

(Article begins on next page)

Sentence Disambiguation by a Shift-Reduce Parsing Technique*

Stuart M. Shieber

Artificial Intelligence Center
SRI International
333 Ravenswood Avenue
Menlo Park, CA 94025

Abstract

Native speakers of English show definite and consistent preferences for certain readings of syntactically ambiguous sentences. A user of a natural-language-processing system would naturally expect it to reflect the same preferences. Thus, such systems must model in some way the *linguistic performance* as well as the *linguistic competence* of the native speaker. We have developed a parsing algorithm—a variant of the LALR(1) shift-reduce algorithm—that models the preference behavior of native speakers for a range of syntactic preference phenomena reported in the psycholinguistic literature, including the recent data on lexical preferences. The algorithm yields the preferred parse deterministically, without building multiple parse trees and choosing among them. As a side effect, it displays appropriate behavior in processing the much discussed garden-path sentences. The parsing algorithm has been implemented and has confirmed the feasibility of our approach to the modeling of these phenomena.

1. Introduction

For natural language processing systems to be useful, they must assign the same interpretation to a given sentence that a native speaker would, since that is precisely the behavior users will expect. Consider, for example, the case of ambiguous sentences. Native speakers of English show definite and consistent preferences for certain readings of syntactically ambiguous sentences [Kimball, 1973, Frazier and Fodor, 1978, Ford *et al.*, 1982]. A user of a natural-language-processing system would naturally expect it to reflect the same preferences. Thus, such systems must model in some way the *linguistic performance* as well as the *linguistic competence* of the native speaker.

This idea is certainly not new in the artificial-intelligence literature. The pioneering work of Marcus [Marcus, 1980] is perhaps the best known example of linguistic-performance modeling in AI. Starting from the hypothesis that "deterministic" parsing of English is possible, he demonstrated that certain performance constraints, e.g., the difficulty of parsing garden-path sentences, could be modeled. His claim about deterministic parsing was quite strong. Not only was the behavior of the parser required to be deterministic, but, as Marcus claimed,

> The interpreter cannot use some general rule to take a nondeterministic grammar specification and impose arbitrary constraints to convert it to a deterministic specification (unless, of course, there is a general rule which will always lead to the correct decision in such a case). [Marcus, 1980, p.14]

We have developed and implemented a parsing system that, given a nondeterministic grammar, forces disambiguation in just the manner Marcus rejected (i.e. through general rules); it thereby exhibits the same preference behavior that psycholinguists have attributed to native speakers of English for a certain range of ambiguities. These include structural ambiguities [Frazier and Fodor, 1978, Frazier and Fodor, 1980, Wanner, 1980] and lexical preferences [Ford *et al.*, 1982], as well as the garden-path sentences as a side effect. The parsing system is based on the shift-reduce scheduling technique of Pereira [forthcoming].

Our parsing algorithm is a slight variant of LALR(1) parsing, and, as such, exhibits the three conditions postulated by Marcus for a deterministic mechanism: it is data-driven, reflects expectations, and has look-ahead. Like Marcus's parser, our parsing system is deterministic. Unlike Marcus's parser, the grammars used by our parser can be ambiguous.

2. The Phenomena to be Modeled

The parsing system was designed to manifest preferences among structurally distinct parses of ambiguous sentences. It does this by building just one parse tree—rather than building multiple parse trees and choosing among them. Like the Marcus parsing system, ours does not do disambiguation requiring "extensive semantic processing," but, in contrast to Marcus, it does handle such phenomena as PP-attachment insofar as there exist *a priori* preferences for one attachment over another. By *a priori* we mean preferences that are exhibited in contexts where pragmatic or plausibility considerations do not tend to favor one reading over the other. Rather than make such value judgments ourselves, we defer to the psycholinguistic literature (specifically [Frazier and Fodor, 1978], [Frazier and Fodor, 1980] and [Ford *et al.*, 1982]) for our examples.

*This research was supported by the Defense Advanced Research Projects Agency under Contract N00039-80-C-0575 with the Naval Electronic Systems Command. The views and conclusions contained in this document are those of the author and should not be interpreted as representative of the official policies, either expressed or implied, of the Defense Advanced Research Projects Agency or the United States government.

The parsing system models the following phenomena:

Right Association

Native speakers of English tend to prefer readings in which constituents are "attached low." For instance, in the sentence

Joe bought the book that I had been trying to obtain for Susan.

the preferred reading is one in which the prepositional phrase "for Susan" is associated with "to obtain" rather than "bought."

Minimal Attachment

On the other hand, higher attachment is preferred in certain cases such as

Joe bought the book for Susan.

in which "for Susan" modifies "the book" rather than "bought." Frazier and Fodor [1978] note that these are cases in which the higher attachment includes fewer nodes in the parse tree. Our analysis is somewhat different.

Lexical Preference

Ford *et al.* [1982] present evidence that attachment preferences depend on lexical choice. Thus, the preferred reading for

The woman wanted the dress on that rack.

has low attachment of the PP, whereas

The woman positioned the dress on that rack.

has high attachment.

Garden-Path Sentences

Grammatical sentences such as

The horse raced past the barn fell.

seem actually to receive no parse by the native speaker until some sort of "conscious parsing" is done. Following Marcus [Marcus, 1980], we take this to be a hard failure of the human sentence-processing mechanism.

It will be seen that all these phenomena are handled in our parser by the same general rules. The simple context-free grammar used[1] (see Appendix I) allows both parses of the ambiguous sentences as well as one for the garden-path sentences. The parser disambiguates the grammar and yields only the preferred structure. The actual output of the parsing system can be found in Appendix II.

3. The Parsing System

The parsing system we use is a shift-reduce parser. Shift-reduce parsers [Aho and Johnson, 1974] are a very general class of bottom-up parsers characterized by the following architecture. They incorporate a *stack* for holding constituents built up during the parse and a *shift-reduce table* for guiding the parse. At each step in the parse, the table is used for deciding between two basic types of operations: the *shift* operation, which adds the next word in the sentence (with its preterminal category) to the top of the stack, and the *reduce* operation, which removes several elements from the top of the stack and replaces them with a new element—for instance, removing an NP and a VP from the top of the stack and replacing them with an S. The *state* of the parser is also updated in accordance with the shift-reduce table at each stage. The combination of the stack, input, and state of the parser will be called *a configuration* and will be notated as, for example,

NP V	*Mary*	10

where the stack contains the nonterminals NP and V, the input contains the lexical item Mary and the parser is in state 10.

By way of example, we demonstrate the operation of the parser (using the grammar of Appendix I) on the oft-cited sentence "John loves Mary." Initially the stack is empty and no input has been consumed. The parser begins in state 0.

	John loves Mary	0

As elements are shifted to the stack, they are replaced by their preterminal category.[2] The shift-reduce table for the grammar of Appendix I states that in state 0, with a proper noun as the next word in the input, the appropriate action is a shift. The new configuration, therefore, is

PNOUN	*loves Mary*	4

The next operation specified is a reduction of the proper noun to a noun phrase yielding

NP	*loves Mary*	2

The verb and second proper noun are now shifted, in accordance with the shift-reduce table, exhausting the input, and the proper noun is then reduced to an NP.

NP V	*Mary*	10
NP V PNOUN		4
NP V NP		14

Finally, the verb and noun phrase on the top of the stack are reduced to a VP

NP VP		6

which is in turn reduced, together with the subject NP, to an S.

S		1

This final configuration is an accepting configuration, since all

[1] We make no claims as to the accuracy of the sample grammar. It is obviously a gross simplification of English syntax. Its role is merely to show that the parsing system is able to disambiguate the sentences under consideration correctly.

[2] But see Section 3.2 for an exception.

the input has been consumed and an S derived. Thus the sentence is grammatical in the grammar of Appendix I, as expected.

3.1 Differences from the Standard LR Techniques

The shift-reduce table mentioned above is generated automatically from a context-free grammar by the standard algorithm [Aho and Johnson, 1974]. The parsing alogrithm differs, however, from the standard LALR(1) parsing algorithm in two ways. First, instead of assigning preterminal symbols to words as they are shifted, the algorithm allows the assignment to be delayed if the word is ambiguous among preterminals. When the word is used in a reduction, the appropriate preterminal is assigned.

Second, and most importantly, since true LR parsers exist only for unambiguous grammars, the normal algorithm for deriving LALR(1) shift-reduce tables yields a table that may specify conflicting actions under certain configurations. It is through the choice made from the options in a conflict that the preference behavior we desire is engendered.

3.2 Preterminal Delaying

One key advantage of shift-reduce parsing that is critical in our system is the fact that decisions about the structure to be assigned to a phrase are postponed as long as possible. In keeping with this general principle, we extend the algorithm to allow the assignment of a preterminal category to a lexical item to be deferred until a decision is forced upon it, so to speak, by an encompassing reduction. For instance, we would not want to decide on the preterminal category of the word "that," which can serve as either a determiner (DET) or complementizer (THAT), until some further information is available. Consider the sentences

That problem is important.

That problems are difficult to solve is important.

Instead of assigning a preterminal to "that," we leave open the possibility of assigning either DET or THAT until the first reduction that involves the word. In the first case, this reduction will be by the rule NP →DET NOM, thus forcing, once and for all, the assignment of DET as preterminal. In the second case, the DET NOM analysis is disallowed on the basis of number agreement, so that the first applicable reduction is the COMP S reduction to $\overline{S}$, forcing the assignment of THAT as preterminal.

Of course, the question arises as to what state the parser goes into after shifting the lexical item "that." The answer is quite straightforward, though its interpretation *vis à vis* the determinism hypothesis is subtle. The simple answer is that the parser enters into a state corresponding to the union of the states entered upon shifting a DET and upon shifting a THAT respectively, in much the same way as the deterministic simulation of a nondeterministic finite automaton enters a "union" state when faced with a nondeterministic choice. Are we then merely simulating a nondeterministic machine here? The answer is equivocal. Although the implementation acts as a simulator for a nondeterministic machine, the nondeterminism is *a priori* bounded, given a particular grammar and lexicon.[3] Thus, the nondeterminism could be traded in for a larger, albeit still finite, set of states, unlike the nondeterminism found in other parsing algorithms. Another way of looking at the situation is to note that there is no observable property of the algorithm that would distinguish the operation of the parser from a deterministic one. In some sense, there is no interesting difference between the limited nondeterminism of this parser, and Marcus's notion of strict determinism. In fact, the implementation of Marcus's parser also embodies a bounded nondeterminism in much the same way this parser does.

The differentiating property between this parser and that of Marcus is a slightly different one, namely, the property of *quasi-real-time operation.*[4] By quasi-real-time operation, Marcus means that there exists a maximum interval of parser operation for which no output can be generated. If the parser operates for longer than this, it must generate some output. For instance, the parser might be guaranteed to produce output (i.e., structure) at least every three words. However, because preterminal assignment can be delayed indefinitely in pathological grammars, there may exist sentences in such grammars for which arbitrary numbers of words need to be read before output can be produced. It is not clear whether this is a real disadvantage or not, and, if so, whether there are simple adjustments to the algorithm that would result in quasi-real-time behavior. In fact, it is a property of bottom-up parsing in general that quasi-real-time behavior is not guaranteed. Our parser has a less restrictive but similar property, *fairness*, that is, our parser generates output linear in the input, though there is no constant over which output is guaranteed. For a fuller discussion of these properties, see Pereira and Shieber [forthcoming].

To summarize, preterminal delaying, as an intrinsic part of the algorithm, does not actually change the basic properties of the algorithm in any observable way. Note, however, that preterminal assignments, like reductions, are irrevocable once they are made (as a byproduct of the determinism of the algorithm). Such decisions can therefore lead to garden paths, as they do for the sentences presented in Section 3.6.

We now discuss the central feature of the algorithm, namely, the resolution of shift-reduce conflicts.

3.3 The Disambiguation Rules

Conflicts arise in two ways: *shift-reduce* conflicts, in which the parser has the option of either shifting a word onto the stack or reducing a set of elements on the stack to a new element; *reduce-reduce* conflicts, in which reductions by several grammar

[3] The boundedness comes about because only a finite amount of information is kept per state (an integer) and the nondeterminism stops at the preterminal level, so that the splitting of states does not propogate.

[4] I am indebted to Mitch Marcus for this observation and the previous comparison with his parser.

rules are possible. The parser uses two rules to resolve these conflicts:[5]

(1) Resolve shift-reduce conflicts by shifting.

(2) Resolve reduce-reduce conflicts by performing the longer reduction.

These two rules suffice to engender the appropriate behavior in the parser for cases of right association and minimal attachment. Though we demonstrate our system primarily with PP-attachment examples, we claim that the rules are generally valid for the phenomena being modeled [Pereira and Shieber, forthcoming].

3.4 Some Examples

Some examples demonstrate these principles. Consider the sentence

Joe took the book that I bought for Susan.

After a certain amount of parsing has been completed deterministically, the parser will be in the following configuration:

NP V NP that NP V	*for Susan*	23

with a shift-reduce conflict, since the V can be reduced to a VP/NP[6] or the P can be shifted. The principles presented would solve the conflict in favor of the shift, thereby leading to the following derivation:

NP V NP that NP V P	*Susan*	12
NP V NP that NP V P NP		19
NP V NP that NP V PP		24
NP V NP that NP VP/NP		22
NP V NP that S/NP		16
NP V NP $\overline{S}$		7
NP V NP		14
NP VP		6
S		1

which yields the structure:

$[_S$ Joe$[_{VP}$took$[_{NP}[_{NP}$the book$][_{\overline{S}}$that I bought for Susan$]]]]$

The sentence

Joe bought the book for Susan.

demonstrates resolution of a reduce-reduce conflict. At some point in the parse, the parser is in the following configuration:

NP V NP PP		20

with a reduce-reduce conflict. Either a more complex NP or a VP can be built. The conflict is resolved in favor of the longer reduction, i.e., the VP reduction. The derivation continues:

NP VP		6
S		1

ending in an accepting state with the following generated structure:

$[_S$ Joe$[_{VP}$ bought$[_{NP}$the book$][_{PP}$for Susan$]]]$

3.5 Lexical Preference

To handle the lexical-preference examples, we extend the second rule slightly. Preterminal-word pairs can be stipulated as either *weak* or *strong*. The second rule becomes

(2) Resolve reduce-reduce conflicts by performing the longest reduction *with the strongest leftmost stack element.*[7]

Therefore, if it is assumed that the lexicon encodes the information that the triadic form of "want" (V2 in the sample grammar) and the dyadic form of "position" (V1) are both weak, we can see the operation of the shift-reduce parser on the "dress on that rack" sentences of Section 2. Both sentences are similar in form and will thus have a similar configuration when the reduce-reduce conflict arises. For example, the first sentence will be in the following configuration:

NP wanted NP PP		20

In this case, the longer reduction would require assignment of the preterminal category V2 to "want," which is the weak form: thus, the shorter reduction will be preferred, leading to the derivation:

NP wanted NP		14
NP VP		6
S		1

and the underlying structure:

$[_S$the woman$[_{VP}$wanted$[_{NP}[_{NP}$the dress$][_{PP}$on that rack$]]]]$

[5] The original notion of using a shift-reduce parser and general scheduling principles to handle right association and minimal attachment, together with the following two rules, are due to Fernando Pereira [Pereira, 1982]. The formalization of preterminal delaying and the extensions to the lexical-preference cases and garden-path behavior are due to the author.

[6] The "slash-category" analysis of long-distance dependencies used here is loosely based on the work of Gazdar [1981]. The Appendix I grammar does not incorporate the full range of slashed rules, however, but merely a representative selection for illustrative purposes.

[7] Note that strength takes precedence over length.

In the case in which the verb is "positioned," however, the longer reduction does not yield the weak form of the verb; it will therefore be invoked, resting in the structure:

[$_S$the woman[$_{VP}$positioned[$_{NP}$the dress][$_{PP}$on that rack]]]

3.6 Garden-Path Sentences

As a side effect of these conflict resolution rules, certain sentences in the language of the grammar will receive no parse by the parsing system just discussed. These sentences are apparently the ones classified as "garden-path" sentences, a class that humans also have great difficulty parsing. Marcus's conjecture that such difficulty stems from a hard failure of the normal sentence-processing mechanism is directly modeled by the parsing system presented here.

For instance, the sentence

The horse raced past the barn fell.

exhibits a reduce-reduce conflict before the last word. If the participial form of "raced" is weak, the finite verb form will be chosen; consequently, "raced past the barn" will be reduced to a VP rather than a participial phrase. The parser will fail shortly, since the correct choice of reduction was not made.

Similarly, the sentence

That scaly, deep-sea fish should be underwater is important.

will fail, though grammatical. Before the word "should" is shifted, a reduce-reduce conflict arises in forming an NP from either "That scaly, deep-sea fish" or "scaly, deep-sea fish." The longer (incorrect) reduction will be performed and the parser will fail.

Other examples, e.g., "the boy got fat melted," or "the prime number few" would be handled similarly by the parser, though the sample grammar of Appendix I does not parse them [Pereira and Shieber, forthcoming].

4. Conclusion

To be useful, natural-language systems must model the behavior, if not the method, of the native speaker. We have demonstrated that a parser using simple general rules for disambiguating sentences can yield appropriate behavior for a large class of performance phenomena—right association, minimal attachment, lexical preference, and garden-path sentences—and that, morever, it can do so deterministically without generating all the parses and choosing among them. The parsing system has been implemented and has confirmed the feasibility of our approach to the modeling of these phenomena.

References

Aho, A.V., and S.C. Johnson, 1974: "LR Parsing," *Computing Surveys*, Volume 6, Number 2, pp. 99-124 (Spring).

Ford, M., J. Bresnan, and R. Kaplan, 1982: "A Competence-Based Theory of Syntactic Closure," in *The Mental Representation of Grammatical Relations*, J. Bresnan, ed. (Cambridge, Massachusetts: MIT Press).

Frazier, L., and J.D. Fodor, 1978: "The Sausage Machine: A New Two-Stage Parsing Model," *Cognition*, Volume 6, pp. 291-325.

Frazier, L., and J.D. Fodor, 1980: "Is the Human Sentence Parsing Mechanism an ATN?" *Cognition*, Volume 8, pp. 411-459.

Gazdar, G., 1981: "Unbounded dependencies and coordinate structure," *Linguistic Inquiry*, Volume 12, pp. 165-179.

Kimball, J., 1973: "Seven Principles of Surface Structure Parsing in Natural Language," *Cognition*, Volume 2, Number 1, pp. 15-47.

Marcus, M., 1980: *A Theory of Syntactic Recognition for Natural Language*, (Cambridge, Massachusetts: MIT Press).

Pereira, F.C.N., forthcoming: "A New Characterization of Attachment Preferences," to appear in D. Dowty, L. Karttunen, and A. Zwicky (eds.) *Natural Language Processing. Psycholinguistic, Computational, and Theoretical Perspectives*, Cambridge, England: Cambridge University Press.

Pereira, F.C.N., and S.M. Shieber, forthcoming: "Shift-Reduce Scheduling and Syntactic Closure," to appear.

Wanner, E., 1980: "The ATN and the Sausage Machine: Which One is Baloney?" *Cognition*, Volume 8, pp. 209–225.

Appendix I. The Test Grammar

The following is the grammar used to test the parsing system descibed in the paper. Not a robust grammar of English by any means, it is presented only for the purpose of establishing that the preference rules yield the correct results.

S → NP VP	VP → V3 INF
S → $\bar{S}$ VP	VP → V4 ADJ
NP → DET NOM	VP → V5 PP
NP → NOM	$\bar{S}$→ that S
NP → PNOUN	INF → to VP
NP → NP $\bar{S}$/NP	PP → P NP
NP → NP PARTP	PARTP → VPART PP
NP → NP PP	$\bar{S}$/NP → that S/NP
DET → NP 's	S/NP → VP
NOM → N	S/NP → NP VP/NP
NOM → ADJ NOM	VP/NP → V1
VP → AUX VP	VP/NP → V2 PP
VP → V0	VP/NP → V3 INF/NP
VP → V1 NP	VP/NP → AUX VP/NP
VP → V2 NP PP	INF/NP → to VP/NP

Appendix II. Sample Runs

```
>> Joe bought the book that I had been trying to obtain
   for Susan
```

```
Accepted: [s
              (np (pnoun Joe))
              (vp
                (v1 bought)
                (np
                  (np (det the)
                      (nom (n book)))
                  (sbar/np
                    (that that)
                    (s/np
                      (np (pnoun I))
                      (vp/np
                        (aux had)
                        (vp/np
                          (aux been)
                          (vp/np (v3 trying)
                                 (inf/np
                                   (to to)
                                   (vp/np
                                     (v2 obtain)
                                     (pp (p for)
                                         (np (pnoun Susan]


>> Joe bought the book for Susan

Accepted: [s (np (pnoun Joe))
               (vp (v2 bought)
                   (np (det the)
                       (nom (n book)))
                   (pp (p for)
                       (np (pnoun Susan]

>> The woman wanted the dress on that rack

Accepted: [s (np (det The)
                 (nom (n woman)))
               (vp (v1 wanted)
                   (np (np (det the)
                           (nom (n dress)))
                       (pp (p on)
                           (np (det that)
                               (nom (n rack]

>> The woman positioned the dress on that rack

Accepted: [s (np (det The)
                 (nom (n woman)))
               (vp (v2 positioned)
                   (np (det the)
                       (nom (n dress)))
                   (pp (p on)
                       (np (det that)
                           (nom (n rack]


>> The horse raced past the barn fell

Parse failed.  Current configuration:
state:  (1)

stack:  <(0)>       [s (np (det The)
                           (nom (n horse)))
                       (vp (v5 raced)
                           (pp (p past)
                               (np (det the)
                                   (nom (n barn]


input:  (v0 fell)
         (end)

>> That scaly deep-sea fish should be underwater is important

Parse failed.  Current configuration:

state:  (1)

stack:  <(0)>       [s [np (det That)
                           (nom (adj scaly)
                                (nom (adj deep-sea)
                                     (nom (n fish]
                       (vp (aux should)
                           (vp (v4 be)
                               (adj underwater]


input:  (v4 is)
         (adj important)
         (end)
```